Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Computer Systems Software
Department

Fundamentals of Programming. Part 1. Basic Constructions

Syllabus

Requisites of the Course

Cycle of Higher Education	***First cycle of higher education (Bachelor's degree)***
Field of Study	*12 Information Technologies*
Speciality	*121 Software engineering*
Education Program	*Software Engineering of Multimedia and Information Retrieval Systems*
Type of Course	*Normative*
Mode of Studies	*full-time*
Year of studies, semester	*1 year (1 semester)*
ECTS workload	*5.5 credits (ECTS). Time allotment - 165 hours, including 90 hours of classroom work, and 75 hours of self-study.*
Testing and assessment	*1 semester – Exam*
Course Schedule	*2(3) classes per week by the timetable http://roz.kpi.ua/*
Language of Instruction	*English*
Course Instructors	Lecturer: PhD, Associate Professor, Yuliia Boiarinova, mobile +380671751308, email ub@ua.fm Teacher of practical work: PhD, Associate Professor, Yuliia Boiarinova, mobile +380671751308, email ub@ua.fm Teacher of laboratory work: PhD, Associate Professor, Yuliia Boiarinova, mobile +380671751308, email ub@ua.fm
Access to the course	https://t.me/+zub7kDn0N2g0Zjc6 https://classroom.google.com/c/NTQ2MDgwNTcwMjA3?cjc=nvsujjp

Outline of the Course

1. Course description, goals, objectives, and learning outcomes

The discipline "Fundamentals of Programming. Part 1. Basic Constructions" is aimed at studying the theoretical and methodological foundations of building programs in the programming language C, mastering the means of creating software, gainingpractical skills in software development in solving practical problems. Such theoretical and practical training forms basic skills in programming and is the basis for successful mastering of professional disciplines.

The purpose *of the discipline is to form students' ability to develop software for solving applied problems of varying complexity in the C programming language.*

Studying the discipline «Fundamentals of Programming. Part 1. Basic Constructions» generates ***general competence (GC) and professional competence (PC):***
GC 01 *Ability to abstract thinking, analysis and synthesis.*
GC 06 *Ability to search, process and analyze information from various sources.*
PC 01 *Ability to identify, classify and formulate software requirements.*
PC 02 *Ability to participate in software design, including its structure, behavior and functioning processes modeling (formal description).*

PC 03 *Ability to develop software systems architectures, modules and components.*
PC07 *Knowledge of information data models, the ability to create software for data storage, retrieval and processing.*
PC 08 *Ability to apply fundamental and interdisciplinary knowledge to successfully solve software engineering problems.*
PC 10 *Ability to accumulate, process and systematize professional knowledge about software creation and maintenance, and determination of the importance of lifelong learning.*
PC 11 *Ability to implement phases and iterations of the life cycle of the software systems and information technology based on appropriate models and approaches to software development.*
PC 13 *Ability to reasonably select and master software development and maintenance tools.*
PC 14 *Ability to algorithmic and logical thinking.*

Programming Learning Outcomes (PLO) *of the discipline «Fundamentals of Programming. Part 1. Basic Constructions» under the educational program:*
PLO 01 *To analyze, purposefully search and select the necessary information and reference resources and knowledge to solve professional problems, taking into account modern advances in science and technology.*
PLO03 *To know the software life cycle basic processes, phases and iterations.*
PLO06 *Ability to select and use the appropriate task of software development methodology.*
PLO07 *To know and to apply in practice the fundamental concepts, paradigms and basic principles of the functioning of language, instrumental and computational tools of software engineering.*
PLO13 *To know and apply methods of developing algorithms, designing software and data and knowledge structures.*
PLO15 *To choose programming languages and development technologies to solve the problems of creating and maintaining software.*
PLO18 *To know and be able to apply information technology of processing, storage and transmission of data.*
PLO38 *To be able to apply programming technologies for multimedia and information retrieval systems software development.*

2. Prerequisites and post-requisites of the course (the place of the course in the scheme of studies in accordance with curriculum)

To successfully master the discipline «Fundamentals of Programming. Part 1. Basic Constructions» it is necessary and sufficient to have training at the secondary school level, basic knowledge of working with a PC, if possible, the basics of programming in the amount provided by the high school program.

To the successful study of the discipline «Fundamentals of Programming. Part 1. Basic Constructions» precedes the study of the disciplines of the school course of mathematics (for the analysis of numerical data, which are described by mathematical laws) and computer science (for means of processing and storing data on a personal computer).

Received during the assimilation of the discipline «Fundamentals of Programming. Part 1. Basic Constructions» theoretical knowledge and practical skills are necessary for most disciplines of the curriculum and the educational program as a whole, in particular, the study of the disciplines "Programming", "Databases", "Software Engineering Components" and other disciplines of the curriculum of undergraduate training in the specialty 121 Software Engineering, as well as for successful completion of pre-diploma practice, course and diploma projects in the specialty 121 Software Engineering.

3. Content of the course

Topic 1. Fundamentals of programming.

Topic 2. C programming language: data types and basic language constructs

Topic 3. Complex data types in C. Arrays

Topic 4. Functions in C.

4. Coursebooks and teaching resources

Basic

1. The C Programming Language/ Kernighan,Brian; Ritchie, Dennis M. - Englewood Cliffs, NJ:Prentice Hall, 1988 -288p.

2. C: How to Program/ Paul J. Deitel, Harvey M. Deitel,Pearson Prentice Hall, 2010 -998p.
Aditional

3. Sibling rivalry: C and C++/ Stroustrup, Bjarne AT&T Labs. Archived from the original on August 24, 2014.

4. https://www.programiz.com/c-programming

5. https://www.tutorialspoint.com/cprogramming/index.htm

6. https://www.cprogramming.com/

Educational content

5. Methodology

№	Type of study	Description of the lesson
I semester		
Topic 1. Fundamentals of programming. Introduction.		
1	Lection 1. History of development of programming languages.	History of development of programming languages. Structure of a computer. Stages solving a problem on acomputer. Self study: item 6, N1
2	Lection 2. Programming languages. Compilers	Characteristics of C-systems. Software developmentenvironment. Compiler structure. Self study: item 6, N2
3	Computer lesson №1	Establishing a software development environment Self study: item 6, N3
4	Practical lesson №1	Description of the algorithm, stages of programdevelopment Self study: item 6, N4
Topic 2. C programming language: data types and basic language constructs		
5	Lection 3. Structure of CProgram.	Introduction to language. Language alphabet, comments. Program structure. Preprocessor directives. Self study: item 6, N5
6	Computer lesson №2	Task: create program and compiling it Self study: item 6, N6
7	Lection 4. Data types	Data types. Type conversion Self study: item 6, N7
8	Computer lesson №3	Task: settings development environment Self study: item 6, N8
9	Practical lesson №2	Linear Algorithms Self study: item 6, N9
10	Lection 5. Data input and output.	Data input and output. Formatting data output Self study: item 6, N10

11	*Computer lesson №4*	*Task: create program – calculate expression by variantusing library math.h* *Self study: item 6, N11*
12	*Lection 6. Language operations C*	*Operations, priority. Operations before (after)increment, before (after) decrement* *Self study: item 6, N12*
13	*Computer lesson №5*	*Task: create program with before (after) increment,before (after) decrement* *Self study: item 6, N13*
14	*Lection 7. Basic language operators C.*	*Basic constructions. Branching operators, conditionaloperations* *Self study: item 6, N14*
15	*Computer lesson №6*	*Task: create program – calculate expression by variantwith conditions* *Self study: item 6, N15*
16	*Practical lesson №3*	*Algorithms with condition.* *Self study: item 6, N16*
17	*Lection 8. Language operators C.*	*Multiple choice operator* *Self study: item 6, N17*
18	*Computer lesson №7*	*Task: create program – calculate expression by variantwith logical operation* *Self study: item 6, N18*
19	*Practical lesson №4*	*Cyclic algorithms.* *Self study: item 6, N19*
20	*Lection 9. Language operators C*	*Cycle operators and program flow control.* *Self study: item 6, N20*
21	*Computer lesson №8*	*Task: create program with multiple choice operator* *Self study: item 6, N21*
22	*Test №1*	*Self study: item 6, N22*
Topic 3. Complex data types in C.		
23	*Lection 10. Arrays*	*One-dimensional arrays. Initialization, input andoutput of arrays* *Self study: item 6, N23*
24	*Computer lesson №9*	*Task: create program – calculate expression by variant* *with cycle operators*
25	*Practical lesson №5*	*Arranging one-dimensional arrays.* *Self study: item 6, N24*
26	*Lection 11. Arrays*	*Two-dimensional arrays. Initialization, input andoutput of arrays* *Self study: item 6, N25*
27	*Computer lesson №10*	*Task: create program – by variant with one-dimensionarrays* *Self study: item 6, N26*
28	*Lection 12. Arrays*	*Algorithms for finding the minimum-maximumelement* *Self study: item 6, N27*
29	*Computer lesson №11*	*Task: create program – by variant with one-dimensionarrays(continue)* *Self study: item 6, N28*

30	*Practical lesson №6*	*Transformation of one-dimensional arrays* *Self study: item 6, N29*
31	*Lection 13. Arrays*	*Array conversion algorithms* *Self study: item 6, N30*
32	*Computer lesson №12*	*Task: create program – by variant with two-dimensionarrays* *Self study: item 6, N31*
33	*Lection 14. Arrays*	*Search algorithms* *Self study: item 6, N32*
34	*Computer lesson №13*	*Task: create program – by variant with two-dimensionarrays(continue)* *Self study: item 6, N33*
35	*Practical lesson №7*	*Search algorithms* *Self study: item 6, N34*
Topic 4. Functions in C.		
36	*Lection 15. Functions in language C.*	*Definition, caling. Formal and factical parameters* *Self study: item 6, N35*
37	*Computer lesson. №14*	*Task: create program by variant with function* *Self study: item 6, N36*
38	*Lection 16. Functions in language C.*	*Methods of parameter transfer. Create and callfunctions without parameters* *Self study: item 6, N37*
39	*Computer lesson №15*	*Task: create program by variant with function (continue)* *Self study: item 6, N38*
40	*Practical lesson №8*	*Matrix operations* *Self study: item 6, N39*
41	*Lection 17. Functions in language C.*	*Recursive functions. Call recursive functions. Depth ofrecursion* *Self study: item 6, N40*
42	*Computer lesson №16*	*Task: create program by variant with function (continue)* *Self study: item 6, N41*
43	*Lection 18 Functions and arrays in language C.*	*Transfer an array as a parameter to a function.* *Self study: item 6, N42*
44	*Computer lesson №17*	*Task: create program by variant with function andarrays* *Self study: item 6, N43*
45	*Practical lesson №9*	*Algorithms with cycles, array using functions* *Self study: item 6, N44*
46	*Exam*	*Self study: item 6, N45*

6. Self-study

The discipline «Fundamentals of Programming. Part 1. Basic structures» is based on independent preparations for classroom on theoreticaland practical topics.

№	*The name of the topic that is submitted for independent study*	*Quantity of hours*	*Sourses*
1	*Preparation for the lecture 1*	*1*	*1, p.7-10; 2, p.2-3*
2	*Preparation for the lecture 2*	*1*	*1, p.11-14;2, p.4-16*

3	Preparing for a computer lesson 1	1.5	https://www.codeblocks.org/ Install software
4	Preparing for a practical lesson 1	1.5	2, p.24
5	Preparation for the lecture 3	1	1 ,p.37-40
6	Preparing for a computer lesson 2	1.5	https://www.codeblocks.org/ Create test program
7	Preparation for the lecture 4	1	1 ,p.37-38
8	Preparing for a computer lesson 3	1,5	https://gcc.gnu.org
9	Preparing for a practical lesson 2	1.5	https://www.khanacademy.org/computing/computer-science/algorithms/intro-to-algorithms/v/what-are-algorithms
10	Preparation for the lecture 5	1	2, p.24-28
11	Preparing for a computer lesson 4	1.5	https://www.codesdope.com/blog/article/important-functions-in-mathh-library-of-c/ Compile test program
12	Preparation for the lecture 6	1	1, p.41-42
13	Preparing for a computer lesson 5	1.5	Create a program
14	Preparing for a practical lesson 3	1.5	1 ,p.189
15	Preparation for the lecture 7	1	1, p.56-57
16	Preparing for a computer lesson 6	1.5	Create a program
17	Preparation for the lecture 8	1	1,58-59, p.190-191
18	Preparing for a computer lesson 7	1.5	Create a program
19	Preparing for a practical lesson 4	1.5	1, p.60-63
20	Preparation for the lecture 9	1	1 ,p.60-65
21	Preparing for a computer lesson 8	1.5	Create a program
22	Preparing fo test	3	Lection 1-9
23	Preparation for the lecture 10	1	1, p. 26-28, 2, p.195-198
24	Preparing for a computer lesson 9	1.5	Create a program
25	Preparing for a practical lesson 5	1.5	2, p.216-217
26	Preparation for the lecture 11	1	1, p.229-253
27	Preparing for a computer lesson 10	1.5	Create a program
28	Preparation for the lecture 12	1	2, p.198-212
29	Preparing for a computer lesson 11	1.5	Create a program
30	Preparing for a practical lesson 6	1.5	2, p.198-212
31	Preparation for the lecture 13	1	2, p.198-212
32	Preparing for a computer lesson 12	1.5	Create a program
33	Preparation for the lecture 14	1	2, p.223-229
34	Preparing for a computer lesson 13	1.5	Create a program
35	Preparing for a practical lesson 7	1.5	2, p.223-229
36	Preparation for the lecture 15	1	1, p.66-71

37	*Preparing for a computer lesson 14*	*1.5*	*Create a program*
38	*Preparation for the lecture 16*	*1*	*1, p.72-73*
39	*Preparing for a computer lesson 15*	*1.5*	*Create a program*
40	*Preparing for a practical lesson 8*	*1.5*	*2, p.229-252*
41	*Preparation for the lecture 17*	*1*	*1, p.84-86*
42	*Preparing for a computer lesson 16*	*1.5*	*Create a program*
43	*Preparation for the lecture 18*	*1*	*2, p.212-216*
44	*Preparing for a computer lesson 17*	*1.5*	*Create a program*
45	*Preparing for a practical lesson 9*	*1.5*	*2, p.152 ,2,p.198-212*
46	*Preparing for a computer lesson 18*	*1*	*Create a program*
47	*Preparation for exam*	*6*	*Lection 1-18*

Policy and Assessment

7. Course policy

• Attendance at lectures is mandatory.
• Attendance at computer lesson can be sporadic and if necessary to protect the work of the computer lesson.
• Rules of conduct in the classroom: activity, respect for those present, turning off the phones.
• Adherence to the policy of academic integrity.
• Rules for the protection of computer work: work must be done according to the option of the student, which is determined by his number in the list of the group

8. Monitoring and grading policy

During the semester, students complete 6 computer works. Maximum number of points for each computer workshop: 7 points.

Points are awarded for:
- quality of laboratory work (computer work): 0-3 points;
- answer during the defense of laboratory work (computer workshop): 0-3 points;
- timely submission of work to the defense: 0-1 points.
-

Performance evaluation criteria:
3 points - the work is done qualitatively, in full;
2 points - the work is done qualitatively, in full, but has shortcomings;
1 point - the work is done in full, but contains minor errors;
0 points - the work is not performed in full, or contains significant errors.

Response evaluation criteria:
3 points - the answer is complete, well-argued;
2 points - in general the answer is correct, but has shortcomings or minor errors;
1 point - there are significant errors in the answer;
0 points - no answer or the answer is incorrect.

Criteria for assessing the timeliness of submission of work to the defense:
1 points - the work is submitted for defense no later than the specified period;

0 points - the work is submitted for defense later than the specified deadline.

Maximum number of points for performing and defending computer workshops:
R1=7 points × 6 lab. works = 42 points

The test consists of 1 practical task. The answer is evaluated by 8 points.
R2=8
8 points - the answer is correct, complete, well-argued;
7- points - the answer is correct, detailed, but not very well reasoned;
5-6 points - in general the answer is correct, but has shortcomings;
3-4 points - there are minor errors in the answer;
1-2 points - there are significant errors in the answer;
0 points - no answer or the answer is incorrect.

The task for the exam consists of 2 questions - 1 theoretical and 2 practical. The answer to theoretical question is evaluated by 10 points, and the answer to the practical question is evaluated by 20 points.

Criteria for evaluating each theoretical question of the test:
9-10 points - the answer is correct, complete, well-argued;
7-8 points - the answer is correct, detailed, but not very well reasoned;
5-6 points - in general the answer is correct, but has shortcomings;
3-4 points - there are minor errors in the answer;
1-2 points - there are significant errors in the answer;
0 points - no answer or the answer is incorrect.

Criteria for evaluating the practical question of the test:
18-20 points - the answer is correct, the calculations are performed in full;
14-17 points - the answer is correct, but not very well supported by calculations;
9-13 points - in general the answer is correct, but has shortcomings;
5-8 points - there are minor errors in the answer;
1-4 points - there are significant errors in the answer;
0 points - no answer or the answer is incorrect.

Maximum number of points for exam:
R3=10 points × 1 theoretical question + 20 points × 2 practical questions = 50 points.
The rating scale for the discipline is equal to:

Rs = R1 + R2 + R3 = 42 point + 8 point + 50point = 100 points.
Calendar control: conducted twice a semester as a monitoring of the current state of compliance with the requirements of the syllabus.

At the first attestation (8th week) the student receives "credited" if his current rating is not less than 15 points (50% of the maximum number of points that a student can receive before the first attestation).
At the second attestation (14th week) the student receives "credited" if his current rating is not less than 25 points (50% of the maximum number of points that a student can receive before the second attestation).

Semester control: exam

Conditions of admission to semester control: with a semester rating at least 30 points and enrollment in all computer works.

The final performance score or the results of the Fail/ Pass Exam are adopted by university grading system as follows:

Score	*Grade*
100-95	*Excellent*
94-85	*Very good*
84-75	*Good*
74-65	*Satisfactory*
64-60	*Sufficient*
Below 60	*Fail*
Course requirements are not met	*Not Graded*

9. Additional information about the course

It is possible to enroll in certificates of distance or online courses on the relevant topic - programming in C.

Syllabus of the course

Is designed by teacher PhD, Associate Professor, Yuliia Boiarinova
Adopted by Computer Systems Software Department (protocol № 12 from 26.04.23)

Approved by the Faculty Board of Methodology (protocol № 10 from 26.05.23)